AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions and listings of claims in the application. Applicants have submitted a new complete claim set showing marked up claims with insertions indicated by underlining and deletions indicated by strikeouts and/or double bracketing.

1. (Currently amended) A method for providing a topology interface for a multimedia processing system, the method comprising:

receiving a plurality of media parameters identifying at least an identifier, a node type, a data type and a duration; and

in response to receiving the plurality of media parameters, creating by a topology application programming interface a topology interface capable of being passed to a media processor as an extensible symbolic representation of an intended media flow based on at least one of the received media parameters.

2. (Original) The method of claim 1 wherein the media parameters include one or more of a GetCacherObject, a GetNodeType, a GetTopoNodeID, a SetProjectStartStop, a GetProjectStartStop, a GetInputCount, a GetOutputCount, a ConnectOut, a GetInput, a GetOutput, a SetOutputPrefType, a GetOutputPrefType, a SetMajorType, a GetMajorType, a CloneFrom, a SetInputCount, a SetOutputCount, a SetStreamDiscardable, a GetStreamDiscardable, a SetOptionalFlag, and a GetOptonalFlag.

3. (Original) The method of claim 1 wherein the media parameters include a SetSourceAndDescriptor method that enables a topology loader to create a media stream based on a descriptor.

4. (Original) The method of claim 1 wherein the node type is a segment topology node type such that any modifications made to the topology to add, remove or connect nodes does not alter input and output nodes.

5. (Original) The method of claim 1 wherein the unique identifier enables sharing and reusing the nodes in a plurality of topologies.

6. (Original) The method of claim 4 wherein the segment topology node type is created via an IMFSegmentTopologyNode : IUnknown interface.

7. (Original) The method of claim 4 wherein the segment topology node type is created via an IMFSegmentTopologyNode : IUnknown interface including one or more of GetSegmentTopology(IMFTopolgy* pTopology), SegmentTopology(IMFTopology** ppTopology), SetDirty(BOOL bDirty), BOOL IsDirty(), BOOL GetActualOutputNode(long lOutputIndex, IMFTopologyNode** ppActualNode, long* plNodeOutputIndex), and BOOL GetActualInputNode(long lInputIndex, IMFTopologyNode** ppActualNode, long* plNodeInputIndex).

Claims 8-18 are canceled.

19. (Currently amended) A method for providing a segment topology node interface for a multimedia processing system, the method comprising:

receiving a first parameter defining one or more connections for the segment topology node;

receiving a second parameter identifying a pointer to a topology to which the segment topology node can connect; and

in response to receiving the first and second parameter, creating by a segment topology node application programming interface the segment topology node interface as part of a topology that is incapable of alteration of input and output nodes to the segment topology node, the segment topology node being separately identifiable.

20. (Original) The method of claim 19 wherein the segment topology node is created by a topology loader operable through one or more of a SetSegmentTopology(IMFTopolgy* pTopology) command, a GetSegmentTopology(IMFTopology** ppTopology) command, a SetDirty(BOOL bDirty) command, a IsDirty() command, a GetActualOutputNode(long lOutputIndex, IMFTopologyNode** ppActualNode, long* plNodeOutputIndex) command and a GetActualInputNode(long lInputIndex, IMFTopologyNode** ppActualNode, long* plNodeInputIndex) command.

21. (Original) The method of claim 20 wherein the IsDirty and the SetDirty commands relate to a dirty flag on the topology that is inside the segment topology node to determine whether the topology requires resolving.

22. (Original) The method of claim 20 wherein the GetActualOutputNode command and the GetActualInputNode command are used to find a base level non-segment node connected to one of an output stream and an input stream at a predetermined index of the segment topology node.

23. (Currently amended) A method for providing an interface for a multimedia processing system, the method comprising:

receiving a media processor parameter related to received media data;

receiving a timeline parameter related to timing of events to occur for performing media processing; and

receiving a topology parameter describing a flow for the received media data; and in response to receiving the media processor, timeline and topology parameters, enabling by an application programming interface a multimedia processing function via an extensible symbolic abstraction of media objects related to one or more of the media processor parameter, the timeline parameter and the topology parameter.

Claims 24-32 are canceled.